Story 1.6: JPA Entities & Repository Adapters

Status: drafted

Story

As a Developer,
I want JPA entities y repository adapters para persistencia,
so that Puedo persistir/recuperar aggregates desde PostgreSQL siguiendo Hexagonal Architecture.

Acceptance Criteria

AC1: JPA Entity Classes Created

Given Domain models (SignatureRequest, SignatureChallenge) existen (Story 1.5)
When Creo JPA entities en `infrastructure/adapter/outbound/persistence/entity/`
Then

- Clase `SignatureRequestEntity` creada con:
  - `@Entity`, `@Table(name = "signature_request")`
  - `@Id private UUID id` (UUIDv7)
  - Fields mapeados a columnas (customer_id, status, created_at, expires_at, signed_at)
  - `@Type(JsonBinaryType.class)` para `transaction_context` JSONB column
  - `@OneToMany(cascade = CascadeType.ALL, orphanRemoval = true)` para challenges
  - `@Type(JsonBinaryType.class)` para `routing_timeline` JSONB column (List)
- Clase `SignatureChallengeEntity` creada con:
  - `@Entity`, `@Table(name = "signature_challenge")`
  - `@Id private UUID id`
  - `@ManyToOne` back-reference a SignatureRequestEntity
  - Fields mapeados (channel_type, provider, status, sent_at, completed_at, error_code)
  - `@Type(JsonBinaryType.class)` para `provider_proof` JSONB column

AC2: Spring Data JPA Repositories Created

Given JPA entities creadas
When Creo JPA repositories en `infrastructure/adapter/outbound/persistence/repository/`
Then

- `SignatureRequestJpaRepository extends JpaRepository<SignatureRequestEntity, UUID>` creado
- Métodos custom queries opcionales:
  - `Optional<SignatureRequestEntity> findByIdWithChallenges(UUID id)` (@EntityGraph para eager loading)
  - `List<SignatureRequestEntity> findByCustomerId(String customerId)`
  - `List<SignatureRequestEntity> findByStatusAndExpiresAtBefore(String status, Instant expiresAt)`

AC3: Entity Mappers (Bidirectional)

Given Domain models y JPA entities existen
When Creo mappers en `infrastructure/adapter/outbound/persistence/mapper/`
Then

- Clase `SignatureRequestEntityMapper` creada con métodos:
  - `SignatureRequestEntity toEntity(SignatureRequest domain)` - Domain → JPA Entity
  - `SignatureRequest toDomain(SignatureRequestEntity entity)` - JPA Entity → Domain
  - `void updateEntity(SignatureRequest domain, SignatureRequestEntity entity)` - Update existing entity
- Clase `SignatureChallengeEntityMapper` creada (similar)
- Mapeo correcto de:
  - Value Objects (Money, TransactionContext) → JSONB serialization (Jackson)
  - Enums (SignatureStatus, ChallengeStatus) → String columns
  - Collections (List, List)
  - Instant timestamps → database timestamp

AC4: Domain Repository Port Interface

Given Domain layer debe permanecer puro
When Creo port interface en `domain/port/outbound/`
Then

- Interface `SignatureRequestRepository` creada con métodos:
    - `SignatureRequest save(SignatureRequest request)` - Save or update
    - `Optional<SignatureRequest> findById(UUID id)` - Find by ID
    - `List<SignatureRequest> findByCustomerId(String customerId)` - Find by customer
    - `List<SignatureRequest> findExpired(Instant cutoffTime)` - Find expired requests
    - `void delete(UUID id)` - Delete by ID (soft delete future)
- NO dependencies on JPA, Spring, Jackson (domain purity)

AC5: Repository Adapter Implementation

Given Domain port interface y JPA repository existen
When Creo adapter en `infrastructure/adapter/outbound/persistence/adapter/`
Then

- Clase `SignatureRequestRepositoryAdapter implements SignatureRequestRepository` creada
- Usa `SignatureRequestJpaRepository` internamente (dependency injection)
- Usa `SignatureRequestEntityMapper` para conversiones
- Implementa todos los métodos del port interface
- Maneja conversiones domain ↔ entity correctamente
- Retorna domain models (NO JPA entities)

AC6: Hibernate JSONB Support Configuration

Given PostgreSQL JSONB columns necesitan custom type
When Configuro Hibernate para JSONB
Then

- Dependency `io.hypersistence:hypersistence-utils-hibernate-63` agregada a pom.xml
- O alternativa: custom `JsonBinaryType` class creada
- `@Type(JsonBinaryType.class)` funciona en JPA entities

- TransactionContext, ProviderResult, List se serializan/deserializan correctamente

AC7: Integration Tests (Testcontainers)

Given JPA entities y repository adapter implementados
When Creo integration test en `test/java/infrastructure/adapter/outbound/persistence/`
Then

- Clase `SignatureRequestRepositoryIntegrationTest` creada con:
  - `@SpringBootTest`, `@Testcontainers`, `@AutoConfigureTestDatabase(replace = NONE)`
  - `@Container PostgreSQLContainer` para base de datos real
  - Test `testSaveAndFindById()` - Save domain model, find by ID, verify round-trip
  - Test `testCascadePersistChallenges()` - Verify challenges cascade save
  - Test `testJsonbSerializationTransactionContext()` - Verify JSONB serialization
  - Test `testUpdateExistingRequest()` - Update request, verify changes persisted
  - Test `testFindByCustomerId()` - Query by customer ID
  - Test `testFindExpired()` - Query expired requests
- Todos los tests passing (0 failures)

AC8: Transactional Behavior

Given Repository adapter usa Spring Data JPA
When Invoco métodos save/delete
Then

- Métodos repository adapter anotados con `@Transactional` (read-only = false para writes)
- Métodos read-only con `@Transactional(readOnly = true)` para performance
- Rollback automático en caso de exception
- Optimistic locking opcional con `@Version` field (future)

AC9: Package Structure (Hexagonal)

Given Hexagonal Architecture enforcement
When Reviso estructura de packages
Then Estructura es:

```
src/main/java/com/bank/signature/
├── domain/
│   ├── model/                    (Story 1.5 - DONE)
│   └── port/
│       └── outbound/
│           └── SignatureRequestRepository.java  (AC4 - port interface)
└── infrastructure/
    └── adapter/
        └── outbound/
            └── persistence/
                ├── entity/
                │   ├── SignatureRequestEntity.java
                │   └── SignatureChallengeEntity.java
                ├── repository/
                │   └── SignatureRequestJpaRepository.java
                ├── mapper/
                │   ├── SignatureRequestEntityMapper.java
                │   └── SignatureChallengeEntityMapper.java
                └── adapter/
                    └── SignatureRequestRepositoryAdapter.java
```

AC10: ArchUnit Tests Updated

Given Domain purity debe mantenerse
When Actualizo `HexagonalArchitectureTest.java`
Then

- Test `domainPortsShouldNotDependOnInfrastructure()` agregado
  - Verifica que `domain.port.outbound` NO depende de JPA/Spring
- Test `infrastructureShouldNotLeakToApplication()` agregado
  - Verifica que JPA entities NO se exponen fuera de infrastructure.adapter.outbound.persistence
- Test `repositoryAdapterShouldImplementDomainPort()` agregado
  - Verifica que adapter implementa port interface correctamente

AC11: Documentation & Examples

Given Story 1.6 implementado
When Actualizo documentación
Then

- **README.md** actualizado con sección "Persistence Layer (JPA)"
  - Package structure diagram
  - Ejemplo de uso del repository

    - JSONB serialization notes
- **CHANGELOG.md** actualizado con Story 1.6 entry
- JavaDoc en `SignatureRequestRepository` port interface (methods documented)
- JavaDoc en `SignatureRequestRepositoryAdapter` (implementation notes)

AC12: Maven Dependencies Added

Given Story 1.6 requiere nuevas dependencies
When Actualizo `pom.xml`
Then Dependencies agregadas:

- `spring-boot-starter-data-jpa` (ya incluido desde Story 1.1)
- `io.hypersistence:hypersistence-utils-hibernate-63` version 3.7.0 (JSONB support)
- O alternativa: crear custom JsonBinaryType sin dependency externa

Tasks / Subtasks

Task 1: Create Domain Repository Port Interface (AC: #4)

- [ ] Create `src/main/java/com/bank/signature/domain/port/outbound/SignatureRequestRepository.java`
    - [ ] Define interface with 5 methods: save, findById, findByCustomerId, findExpired, delete
    - [ ] Add JavaDoc with @param, @return documentation
    - [ ] NO dependencies on JPA/Spring/Jackson (domain purity)
- [ ] Create `src/main/java/com/bank/signature/domain/port/outbound/package-info.java`
    - [ ] Package documentation explaining outbound ports pattern

Task 2: Add Maven Dependencies (AC: #12)

- [ ] Update `pom.xml`
    - [ ] Add `io.hypersistence:hypersistence-utils-hibernate-63` version 3.7.0
    - [ ] Or implement custom JsonBinaryType class (if avoiding external dependency)
    - [ ] Verify `spring-boot-starter-data-jpa` already present (Story 1.1)

Task 3: Create JPA Entity Classes (AC: #1)

- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/entity/SignatureRequestEntity.java`
  - ◯ Add @Entity, @Table(name = "signature_request") annotations
  - ◯ Add @Id UUID id field
  - ◯ Add all fields matching database schema (customer_id, status, created_at, expires_at, signed_at)
  - ◯ Add @Type(JsonBinaryType.class) for transaction_context JSONB
  - ◯ Add @OneToMany(cascade = ALL, orphanRemoval = true) for challenges
  - ◯ Add @Type(JsonBinaryType.class) for routing_timeline JSONB
  - ◯ Add constructor, getters, setters (or Lombok @Data if preferred)
- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/entity/SignatureChallengeEntity.java`
  - ◯ Add @Entity, @Table(name = "signature_challenge") annotations
  - ◯ Add @Id UUID id field
  - ◯ Add @ManyToOne for signature_request_id foreign key
  - ◯ Add all fields matching database schema
  - ◯ Add @Type(JsonBinaryType.class) for provider_proof JSONB
  - ◯ Add constructor, getters, setters

Task 4: Create Spring Data JPA Repositories (AC: #2)

- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/repository/SignatureRequestJpaRepository.java`
  - ◯ Extend `JpaRepository<SignatureRequestEntity, UUID>`
  - ◯ Add custom query methods:
    - ◯ `Optional<SignatureRequestEntity> findByIdWithChallenges(UUID id)` with @EntityGraph
    - ◯ `List<SignatureRequestEntity> findByCustomerId(String customerId)`
    - ◯ `List<SignatureRequestEntity> findByStatusAndExpiresAtBefore(String status, Instant expiresAt)`

Task 5: Create Entity Mappers (AC: #3)

- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/mapper/SignatureRequestEntityMapper.java`
  - ◯ Implement `toEntity(SignatureRequest domain)` method
    - ◯ Map domain fields to entity fields
    - ◯ Serialize TransactionContext to JSONB (Jackson ObjectMapper)
    - ◯ Serialize List to JSONB
    - ◯ Map enums to String
    - ◯ Map challenges collection (cascade)
  - ◯ Implement `toDomain(SignatureRequestEntity entity)` method
    - ◯ Map entity fields to domain fields
    - ◯ Deserialize JSONB to TransactionContext
    - ◯ Deserialize JSONB to List
    - ◯ Map String to enums
    - ◯ Map challenges collection
  - ◯ Implement `updateEntity(SignatureRequest domain, SignatureRequestEntity entity)` method
    - ◯ Update mutable fields only (status, signed_at, challenges)
  - ◯ Add ObjectMapper @Autowired for JSON serialization
- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/mapper/SignatureChallengeEntityMapper.java`
  - ◯ Similar methods for SignatureChallenge

Task 6: Create Repository Adapter (AC: #5)

- ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/adapter/SignatureRequestRepositoryAdapter.java`
  - ◯ Implement `SignatureRequestRepository` domain port interface
  - ◯ Add @Component annotation (Spring managed bean)
  - ◯ Inject `SignatureRequestJpaRepository` via constructor
  - ◯ Inject `SignatureRequestEntityMapper` via constructor

    - ◯ Implement `save(SignatureRequest)` method
        - ◯ Map domain to entity
        - ◯ Call jpaRepository.save()
        - ◯ Map entity back to domain
    - ◯ Implement `findById(UUID)` method
        - ◯ Call jpaRepository.findById()
        - ◯ Map Optional to Optional
    - ◯ Implement `findByCustomerId(String)` method
    - ◯ Implement `findExpired(Instant)` method
        - ◯ Call jpaRepository.findByStatusAndExpiresAtBefore()
    - ◯ Implement `delete(UUID)` method
    - ◯ Add @Transactional annotations (read-only = false for writes, true for reads)

Task 7: Configure Hibernate JSONB Support (AC: #6)

- ◯ Option A: Use hypersistence-utils
    - ◯ Verify dependency added in pom.xml
    - ◯ Use `@Type(JsonBinaryType.class)` in JPA entities
- ◯ Option B: Create custom JsonBinaryType
    - ◯ Create `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/type/JsonBinaryType.java`
    - ◯ Implement Hibernate UserType interface
    - ◯ Handle PostgreSQL JSONB column type
    - ◯ Use Jackson ObjectMapper for serialization/deserialization

Task 8: Create Integration Tests (AC: #7)

- ◯ Create `src/test/java/com/bank/signature/infrastructure/adapter/outbound/persistence/SignatureRequestRepositoryIntegrationTest.java`
    - ◯ Add @SpringBootTest, @Testcontainers, @AutoConfigureTestDatabase(replace = NONE)
    - ◯ Add @Container PostgreSQLContainer static field
    - ◯ Test `testSaveAndFindById()`
        - ◯ Create SignatureRequest domain model with builder

        - ◯ Save via repository adapter
        - ◯ Find by ID
        - ◯ Assert domain model fields match
    - ◯ Test `testCascadePersistChallenges()`
        - ◯ Create SignatureRequest with 2 challenges
        - ◯ Save via repository adapter
        - ◯ Find by ID
        - ◯ Assert 2 challenges persisted
    - ◯ Test `testJsonbSerializationTransactionContext()`
        - ◯ Create SignatureRequest with complex TransactionContext
        - ◯ Save and reload
        - ◯ Assert TransactionContext deserialized correctly
    - ◯ Test `testUpdateExistingRequest()`
        - ◯ Save request
        - ◯ Update status to SIGNED
        - ◯ Save again
        - ◯ Find by ID
        - ◯ Assert status updated
    - ◯ Test `testFindByCustomerId()`
        - ◯ Save 2 requests for customer A, 1 for customer B
        - ◯ Query by customer A ID
        - ◯ Assert 2 requests returned
    - ◯ Test `testFindExpired()`
        - ◯ Save 1 expired request (expiresAt in past)
        - ◯ Save 1 active request (expiresAt in future)
        - ◯ Query findExpired(Instant.now())
        - ◯ Assert only expired request returned

Task 9: Update ArchUnit Tests (AC: #10)

- ◯ Update `src/test/java/com/bank/signature/HexagonalArchitectureTest.java`
    - ◯ Add test `domainPortsShouldNotDependOnInfrastructure()`
        - ◯ Rule: classes in "..domain.port.." should not depend on JPA/Spring

- ◯ Add test `infrastructureShouldNotLeakToApplication()`
    - ◯ Rule: JPA entities should not be accessed outside persistence package
- ◯ Add test `repositoryAdapterShouldImplementDomainPort()`
    - ◯ Rule: classes named "*RepositoryAdapter" should implement domain port interface

Task 10: Update Documentation (AC: #11)

- ◯ Update `README.md`
    - ◯ Add "Persistence Layer (JPA)" section after "Domain Models"
    - ◯ Include package structure diagram
    - ◯ Include example usage of repository adapter
    - ◯ Note JSONB serialization (TransactionContext, ProviderResult, etc.)
- ◯ Update `CHANGELOG.md`
    - ◯ Add Story 1.6 entry under [Unreleased]
    - ◯ List features: JPA entities, repository adapter, JSONB support, 6 integration tests
- ◯ Add JavaDoc to `SignatureRequestRepository` interface
    - ◯ Document each method with @param, @return
- ◯ Add JavaDoc to `SignatureRequestRepositoryAdapter` class
    - ◯ Implementation notes, transaction behavior

Implementation Highlights

Hexagonal Architecture Pattern

- **Domain Port (Outbound)**: `SignatureRequestRepository` interface in `domain/port/outbound/`
    - Pure domain interface, NO infrastructure dependencies
    - Defines contract for persistence operations
- **Infrastructure Adapter**: `SignatureRequestRepositoryAdapter` in `infrastructure/adapter/outbound/persistence/adapter/`
    - Implements domain port interface
    - Uses Spring Data JPA repository internally
    - Maps domain models ↔ JPA entities via mapper
- **Benefit**: Domain layer remains pure, infrastructure can be swapped (e.g., MongoDB adapter)

JPA Entity Design

- **SignatureRequestEntity**: Root entity with `@OneToMany` challenges
- **SignatureChallengeEntity**: Child entity with `@ManyToOne` back-reference
- **Cascade ALL**: Challenges persist/update/delete with parent
- **orphanRemoval = true**: Removed challenges deleted from database

JSONB Serialization Strategy

- **Hypersistence Utils**: `@Type(JsonBinaryType.class)` for PostgreSQL JSONB columns
- **Jackson ObjectMapper**: Automatic serialization of Value Objects (TransactionContext, ProviderResult, Money)
- **List**: Serialized as JSONB array in `routing_timeline` column

Mapper Pattern (Manual vs MapStruct)

- **Manual Mapping** (Story 1.6): Simple, explicit, no compile-time code generation
- **MapStruct** (Future): Compile-time mapper generation, better performance, less boilerplate
- **Choice**: Manual mapping for Story 1.6 (keep it simple), consider MapStruct in future refactoring

Transactional Behavior

- **@Transactional**: Repository adapter methods
  - `save()`: read-only = false (default)
  - `findById()`, `findByCustomerId()`: read-only = true (optimization)
- **Rollback**: Automatic rollback on RuntimeException
- **Isolation Level**: Default (READ_COMMITTED for PostgreSQL)

Testing Strategy

Integration Tests (Testcontainers)

- **PostgreSQL Container**: Real PostgreSQL 15 database in Docker
- **LiquidBase Auto-Run**: Database schema created automatically on startup
- **Round-Trip Validation**: Save domain model → Find by ID → Assert equals
- **JSONB Validation**: Verify complex objects (TransactionContext, List) serialize/deserialize correctly
- **Cascade Validation**: Verify challenges persist automatically with parent

Target Coverage: > 80% line coverage for persistence package

Source Tree (Files to Create/Modify)

Files to Create (13 files)

Domain Port Interface (1 file):

- `src/main/java/com/bank/signature/domain/port/outbound/SignatureRequestRepository.java`

JPA Entities (2 files):

- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/entity/SignatureRequestEntity.java`
- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/entity/SignatureChallengeEntity.java`

JPA Repository (1 file):

- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/repository/SignatureRequestJpaRepository.java`

Mappers (2 files):

- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/mapper/SignatureRequestEntityMapper.java`
- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/mapper/SignatureChallengeEntityMapper.java`

Repository Adapter (1 file):

- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/adapter/SignatureRequestRepositoryAdapter.java`

Integration Tests (1 file):

- `src/test/java/com/bank/signature/infrastructure/adapter/outbound/persistence/SignatureRequestRepositoryIntegrationTest.java`

Optional - Custom JSONB Type (1 file):

- `src/main/java/com/bank/signature/infrastructure/adapter/outbound/persistence/type/JsonBinaryType.java` (if not using hypersistence-utils)

Files to Modify (4 files)

- `pom.xml` - Add hypersistence-utils dependency
- `src/test/java/com/bank/signature/HexagonalArchitectureTest.java` - Add 3 new tests
- `README.md` - Add "Persistence Layer (JPA)" section
- `CHANGELOG.md` - Add Story 1.6 entry

References to Existing Documentation

- **Story 1.2**: `docs/sprint-artifacts/1-2-postgresql-database-setup-liquidbase-changesets.md` (Database schema)
- **Story 1.5**: `docs/sprint-artifacts/1-5-domain-models-aggregates-entities.md` (Domain models)
- **Architecture**: `docs/architecture/02-hexagonal-structure.md` (Hexagonal patterns)
- **Database Schema**: `docs/architecture/03-database-schema.md` (Table definitions, JSONB columns)
- **Tech Spec Epic 1**: `docs/sprint-artifacts/tech-spec-epic-1.md` (Technology stack)

Definition of Done

- [ ] All 12 Acceptance Criteria verified
- [ ] Domain port interface `SignatureRequestRepository` created (5 methods)
- [ ] 2 JPA entities created (SignatureRequestEntity, SignatureChallengeEntity)
- [ ] 1 Spring Data JPA repository created (SignatureRequestJpaRepository)
- [ ] 2 entity mappers created (bidirectional domain ↔ entity)
- [ ] 1 repository adapter created (implements domain port)
- [ ] JSONB support configured (Hypersistence Utils or custom type)
- [ ] 6 integration tests created (Testcontainers PostgreSQL) with > 80% coverage
- [ ] Transactional behavior configured (@Transactional annotations)
- [ ] Package structure follows Hexagonal Architecture
- [ ] 3 ArchUnit tests added (domain purity, no leakage)
- [ ] Maven dependency added (hypersistence-utils)
- [ ] README.md updated with "Persistence Layer" section
- [ ] CHANGELOG.md updated with Story 1.6 entry
- [ ] JavaDoc added to port interface and adapter

- [ ] Integration tests passing (0 failures)
- [ ] ArchUnit tests passing (domain purity maintained)
- [ ] Code review approved

Dev Agent Record

Context Reference

- `docs/sprint-artifacts/1-6-jpa-entities-repository-adapters.context.xml` (to be created)

Agent Model Used

Claude Sonnet 4.5

Debug Log References

Completion Notes List

File List

Created:

Modified:

Deleted:

Change Log

Date	Author	Change
2025-11-27	BMAD SM Agent	Story 1.6 draft created: JPA Entities & Repository Adapters (Hexagonal persistence)